Calculator de buzunar

Introducere

NUME: DEONISE Costin-Alexandru
GRUPA: 332CC

Proiectul “Calculator de buzunar” este un dispozitiv calculator construit cu ajutorul platformei de dezvoltare Arduino, care poate efectua operații matematice de bază precum adunarea, scăderea, înmulțirea și împărțirea a două numere. Scopul proiectului este de a arăta cum Arduino poate fi folosit pentru a construi dispozitive electronice utile, precum și pentru a încuraja oamenii să își îmbunătățească abilitățile în domeniul programării și ingineriei.

Ideea acestui proiect a pornit de la dorința de a crea un calculator simplu, portabil și ușor de utilizat, care să poată fi construit cu ajutorul componentelor electronice disponibile și accesibile. În plus, proiectul a fost conceput pentru a arăta cum Arduino poate fi utilizat pentru a crea dispozitive utile și practice.

Proiectul este util pentru alții pentru că poate fi folosit ca un instrument educațional pentru a învăța concepte de bază în programare și inginerie electronică. De asemenea, poate fi utilizat ca un dispozitiv portabil pentru efectuarea rapidă a calculelor de bază în diverse situații.

Descriere generală

Proiectul “Calculator de buzunar” este format dintr-un hardware și un software care lucrează împreună pentru a efectua operațiile matematice de bază. Scopul său este de a permite utilizatorului să efectueze operații matematice de bază, cum ar fi adunare, scădere, înmulțire și împărțire, folosind o tastatură numerică și un display LCD.

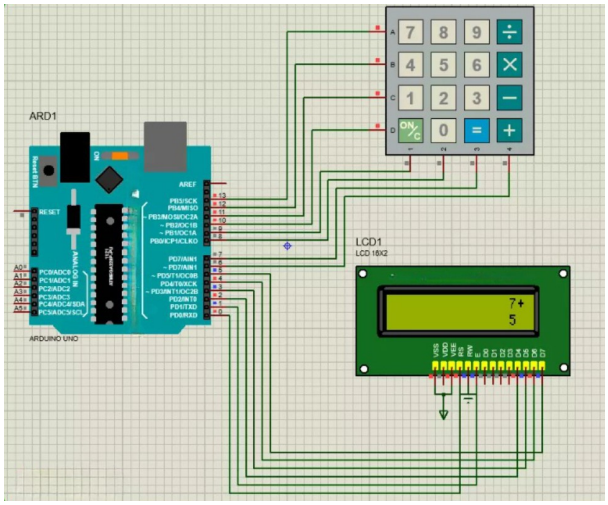
Componentele hardware utilizate în proiect sunt Arduino Uno, un display LCD I2C și o tastatură numerică 4×4. Arduino Uno acționează ca și creierul proiectului, preluând intrările de la tastatură, efectuând calculele și afișând rezultatul pe display-ul LCD.

Tastatura numerică este conectată la Arduino Uno prin intermediul unor pini digitali, iar display-ul LCD este conectat utilizând protocolul I2C, care permite transferul de date pe mai puține fire de conexiune.

Funcționalitatea calculatorului este implementată în codul Arduino. Codul citeste tastatura numerică și efectuează operațiile corespunzătoare pe numerele introduse. Rezultatul este afișat pe display-ul LCD, iar utilizatorul poate continua să introducă și să efectueze noi operații.

Proiectul combină atât partea de hardware, prin utilizarea tastaturii și a display-ului LCD, cât și partea de software, prin implementarea logicii calculatorului în codul Arduino.

În concluzie, acest proiect oferă o soluție simplă și interactivă pentru efectuarea operațiilor matematice de bază, folosind o tastatură numerică și un display LCD conectate la platforma Arduino. Prin combinarea hardware-ului și software-ului, proiectul “Arduino Calculator” poate efectua operații matematice de bază și poate fi utilizat ca un dispozitiv calculator portabil și practic.



Hardware Design

Proiectul de hardware pentru acest calculator simplu constă în utilizarea următoarelor componente:

1. Arduino Uno: Este placa de dezvoltare pe care se bazează proiectul. Arduino Uno este echipat cu un microcontroler ATmega328P și oferă numeroase pini de intrare/ieșire digitali și analogici pentru conectarea și controlul componentelor externe.
2. Display LCD: Se utilizează un display LCD alfanumeric, care afișează numerele și rezultatele operațiilor. Acesta este conectat la Arduino Uno utilizând protocolul I2C (Inter-Integrated Circuit), ceea ce permite transferul datelor pe doar două linii de conexiune.
3. Tastatură numerică: Se utilizează o tastatură numerică matriceală 4×4, care oferă butoane pentru cifrele de la 0 la 9, precum și pentru operațiile matematice (adunare, scădere, înmulțire, împărțire sau paranteză rotundă deschisă, paranteză rotundă închisă, sin și cos). Tastatura este conectată la Arduino Uno prin intermediul piniilor digitali.

Conectori și cabluri: Pentru conectarea componentelor între ele și la Arduino Uno, sunt utilizate fire jumper și conectori adecvați. Acestea asigură conexiunea electrică corectă și stabilă între componentele proiectului.

În ceea ce privește designul hardware, se poate utiliza o placă de conexiune sau o placă de prototipare pentru a organiza componentele într-o configurație ordonată. Arduino Uno este plasat în centrul designului, cu display-ul LCD montat deasupra sau deasupra lui, pentru a permite utilizatorului să vadă rezultatele afișate. Tastatura numerică poate fi plasată într-o poziție convenabilă pentru a permite introducerea ușoară a numerelor și operațiilor.

De asemenea, este important să se ia în considerare alimentarea proiectului. Arduino Uno poate fi alimentat fie prin intermediul unui cablu USB conectat la un computer sau printr-un adaptor de alimentare extern.

În final, designul hardware al acestui calculator simplu implică organizarea componentelor (Arduino

Uno, display LCD, tastatură numerică) într-un mod practic și accesibil, astfel încât utilizatorul să poată interacționa ușor cu calculatorul și să vadă rezultatele afișate pe display.

Software Design

Codul este pentru o simplă calculatoare care poate efectua operații de bază precum adunare, scădere, înmulțire, împărțire, sinus și cosinus. Codul utilizează o tastatură pentru introducerea numerelor și operatorilor și un afișaj LCD pentru a afișa intrarea și rezultatul.

Mediul de dezvoltare specific este Arduino IDE 2.1.0. Se utilizează bibliotecile “Keypad.h”, “LiquidCrystal_I2C.h” și “StackArray.h” pentru funcționalitățile cheii, afișajului LCD și stiva de numere. Aceste biblioteci pot fi biblioteci terțe sau pot face parte dintr-un set de biblioteci personalizate.

Algoritmul utilizat pentru evaluarea expresiei este algoritmul Shunting Yard, care se bazează pe o abordare bazată pe stivă. Expresia este evaluată folosind o stivă de numere și o stivă de operatori. În timpul evaluării, se aplică reguli de precedență pentru a determina ordinea corectă a operațiilor. Algoritmul parsează fiecare caracter al expresiei și îl evaluează în funcție de tipul său (număr, operator sau funcție trigonometrică). Rezultatul final este returnat ca rezultatul expresiei evaluate.

Codul include, de asemenea, funcții auxiliare pentru inițializarea și afișarea rezultatului pe afișajul LCD.

Funcția loop() este bucla principală a programului și se execută în mod repetat. Aceasta gestionează intrarea și ieșirea calculatorului.

1. key = kpd.getKey(): Această instrucțiune primește valoarea tastei apăsate pe tastatură și o stochează în variabila key.
2. if (key != NO_KEY) DetectButtons();: Această condiție verifică dacă a fost apăsată o tastă pe tastatură. Dacă da, se apelează funcția DetectButtons() pentru a trata apăsarea tastelor.
3. if (result == true) answer = evaluateExpression(inputExpression);: Această condiție verifică dacă rezultatul calculului este disponibil. Dacă da, se apelează funcția evaluateExpression(inputExpression) pentru a evalua expresia introdusă și se stochează rezultatul în variabila answer.
4. DisplayResult(): Această instrucțiune afișează rezultatul pe ecranul LCD, utilizând funcția DisplayResult().

Funcția DetectButtons() detectează și gestionează apăsările tastelor în program. Iată o descriere mai concisă a funcționării:

1. Se șterge conținutul ecranului LCD.
2. Se verifică apăsarea butonului “Shift”. Dacă este apăsat, se schimbă starea butonului și se afișează funcțiile asociate pe LCD.
3. Se verifică apăsarea numerelor de la 0 la 9. Numărul apăsat este adăugat la expresia de intrare.
4. Se verifică apăsarea butonului ”#” (egal). Se salvează ultimul număr apăsat și se setează un indicator de rezultat.
5. Se verifică apăsarea butoanelor A, B, C, D în funcție de starea butonului “Shift”. Corespunzător butonului apăsat, se adaugă operatori sau funcții în expresia de intrare.
6. Dacă butonul ”#” este apăsat de două ori consecutiv, se șterge conținutul ecranului LCD și se resetează expresia și numărul.

7. Numărul curent se resetează la zero după adăugarea sa în expresia de intrare, cu excepția cazului în care este apăsat butonul ”#”.

Funcția evaluateExpression() evaluează o expresie matematică dată și returnează rezultatul. Iată o descriere concisă a funcționării:

1. Se declară două stive: numberStack pentru a stoca numere și operatorStack pentru a stoca operatori.
2. Se parcurge fiecare caracter din expresie.
3. Dacă caracterul este o cifră sau punctul zecimal, se construiește numărul curent prin multiplicarea numărului anterior cu 10 și adăugarea cifrei curente.
4. Dacă caracterul nu este o cifră, se verifică tipul acestuia.
5. Dacă este '(', se adaugă la stiva operatorStack.
6. Dacă este ')', se evaluează vârful stivelor numberStack și operatorStack până când se întâlnește '(' și se elimină '(' din stiva operatorStack.
7. Dacă este '+', '-', '*', '/' se evaluează vârful stivelor numberStack și operatorStack în funcție de precedența operatorului curent.
8. Dacă este 's' sau 'c', se adaugă la stiva operatorStack (pentru funcțiile sin și cos).
9. Dacă ultimul caracter evaluat este o cifră, se adaugă numărul final la stiva numberStack.
10. Se evaluează restul operatorilor din stiva operatorStack prin apelul funcției evaluateTop().
11. Dacă stiva numberStack este goală, se returnează 0.0 (expresia era invalidă sau goală).
12. Se returnează rezultatul final, care este vârful stivei numberStack.

Funcția evaluateTop() evaluează vârful stivelor numberStack și operatorStack și realizează operații aritmetice sau calculează funcțiile trigonometrice. Iată o descriere concisă a funcționării:

1. Se declară o variabilă result pentru a stoca rezultatul evaluării și se extrage operatorul din stiva operatorStack prin apelul funcției pop().
2. Dacă operatorul este '+', '-', '*', '/' se extrag cele două numere din stiva numberStack și se realizează operația aritmetică corespunzătoare. Rezultatul este stocat în variabila result.
3. Dacă operatorul este 's', se extrage numărul din vârful stivei numberStack și se calculează sinusul acestuia (convertind gradele în radiani). Rezultatul este stocat în variabila result.
4. Dacă operatorul este 'c', se extrage numărul din vârful stivei numberStack și se calculează cosinusul acestuia (convertind gradele în radiani). Rezultatul este stocat în variabila result.
5. Rezultatul calculat este adăugat înapoi în stiva numberStack prin apelul funcției push().

Rezultate Obţinute

În urma realizării proiectului, s-au obținut următoarele rezultate:

1. S-a implementat un calculator simplu care poate efectua operații de bază precum adunare, scădere, înmulțire, împărțire, sin și cos.
2. Calculatorul utilizează un keypad pentru introducerea numerelor și operatorilor, precum și un ecran LCD pentru afișarea intrării și a rezultatului.
3. Codul este împărțit în trei funcții principale: setup(), loop() și evaluateExpression().
4. Funcția setup() inițializează keypad-ul și ecranul LCD.
5. Funcția loop() este apelată în mod repetat și se ocupă de intrarea și ieșirea calculatorului.
6. Funcția evaluateExpression() evaluează expresia introdusă și returnează rezultatul.

7. S-au utilizat bibliotecile Keypad, LiquidCrystal_I2C și StackArray pentru funcționalitățile specifice.
8. Calculatorul poate manipula numere întregi și numere cu zecimale.
9. S-a implementat algoritmul shunting-yard și o abordare bazată pe stivă pentru evaluarea expresiilor matematice.
10. Operatorii și funcțiile sunt evaluate în funcție de precedența lor.
11. Expresiile matematice pot include paranteze și funcții trigonometrice.
12. Rezultatul este afișat pe ecranul LCD.

Concluzii

Proiectul “Calculator de buzunar” implementează o aplicație simplă de calculator capabilă să efectueze operații de bază precum adunare, scădere, înmulțire, împărțire, sinus și cosinus. Aceasta utilizează un keypad pentru introducerea numerelor și operatorilor, iar rezultatele sunt afișate pe un display LCD.

Codul este structurat în trei funcții principale:

setup(): Această funcție initializează keypad-ul și display-ul LCD. loop(): Această funcție este apelată în mod repetat și se ocupă de intrarea și ieșirea calculatorului. evaluateExpression(): Această funcție evaluează expresia introdusă și returnează rezultatul. Proiectul folosește bibliotecile Keypad și LiquidCrystal_I2C pentru gestionarea keypad-ului și a display-ului LCD, respectiv biblioteca StackArray pentru a implementa o stivă de numere.

Funcția DetectButtons() se ocupă de gestionarea apăsărilor de butoane de pe keypad, inclusiv operatorii, cifrele și funcțiile trigonometrice. Ea actualizează expresia introdusă și setează flag-uri corespunzătoare pentru evaluarea ulterioară a expresiei și afișarea rezultatului.

Funcția evaluateExpression() evaluează expresia introdusă utilizând algoritmul Shunting Yard și o abordare bazată pe stivă. Aceasta descompune expresia în numere și operatori, apoi efectuează operațiile corespunzătoare pe stivă pentru a calcula rezultatul.

Funcțiile precedence() și evaluateTop() ajută la evaluarea expresiei și la efectuarea operațiilor matematice sau trigonometrice.

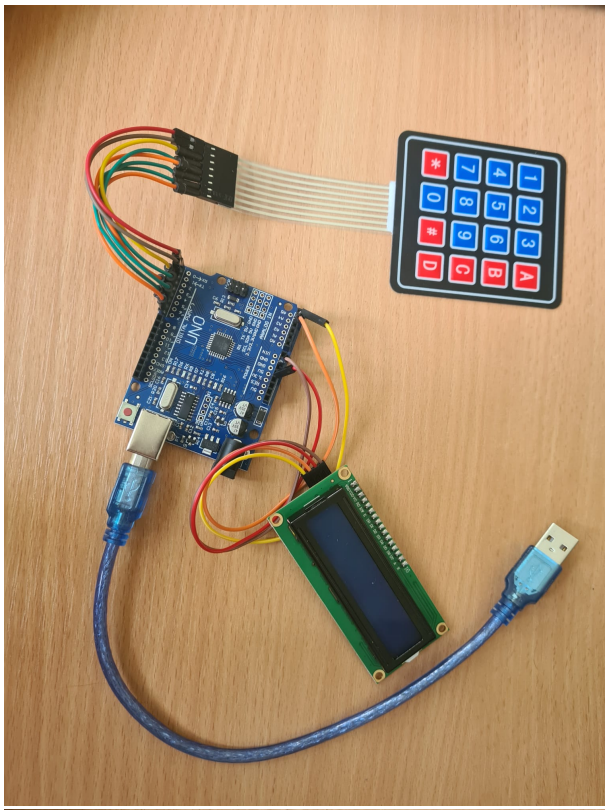
Funcția DisplayResult() se ocupă de afișarea rezultatului pe display-ul LCD, inclusiv expresia introdusă și rezultatul calculat.

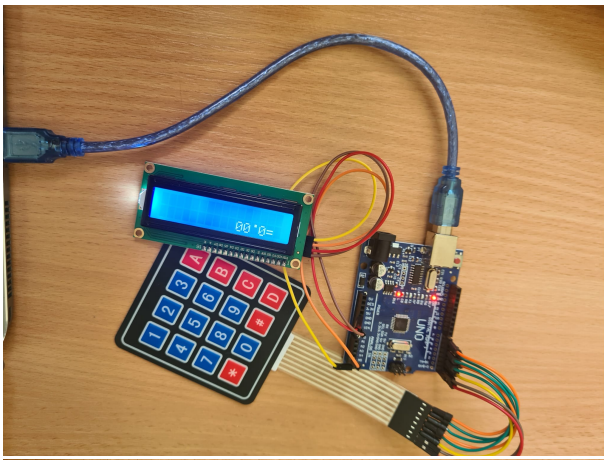
În concluzie, acest proiect implementează un calculator simplu de buzunar care poate efectua operații matematice de bază și funcții trigonometrice. Este o aplicație practică și utilă pentru utilizatorii care doresc să efectueze calcule rapide și ușoare. Cu toate acestea, acest calculator are o funcționalitate limitată și nu suportă operații avansate sau funcții matematice complexe.

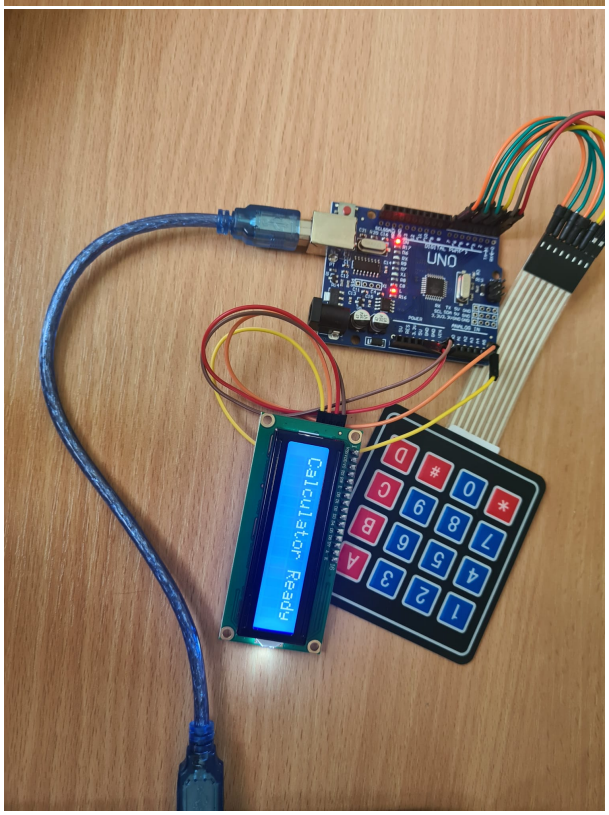
Download

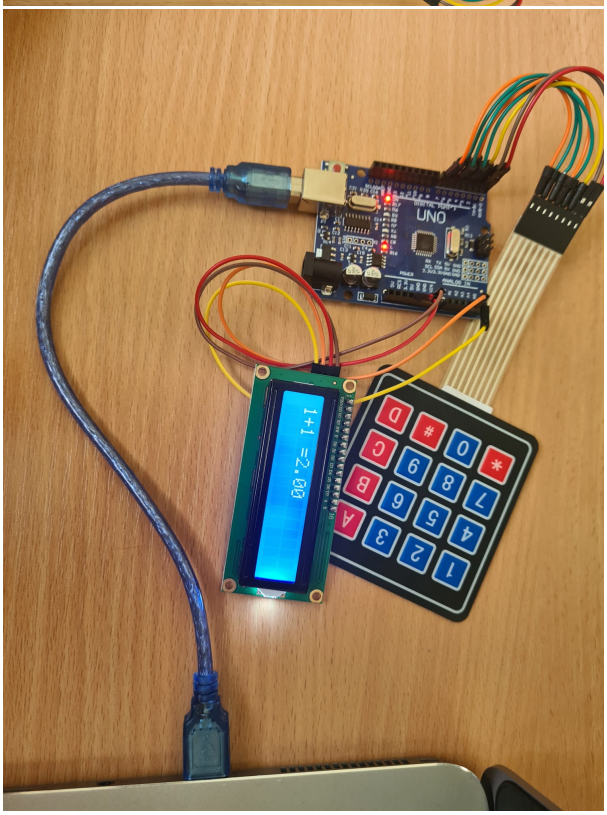
deonise_alex_332cc.zip

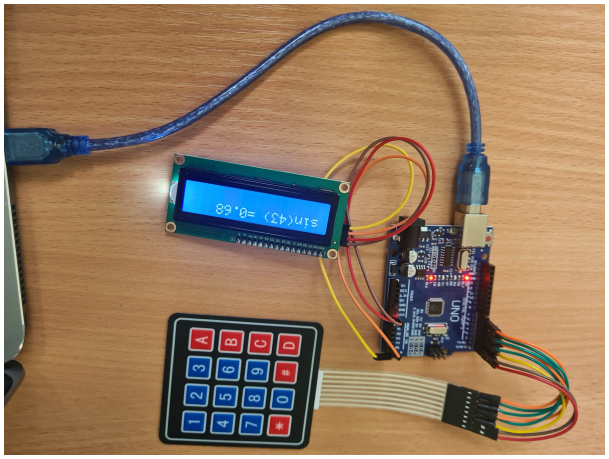
Jurnal

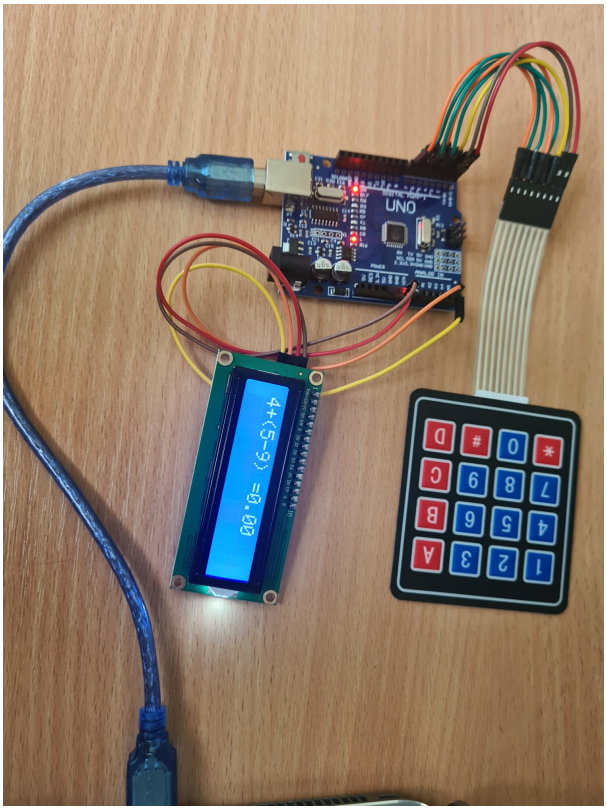












Bibliografie/Resurse

Resurse Software

Bibliotecile necesare:

- Keypad: https://www.arduino.cc/reference/en/libraries/keypad/
- LiquidCrystal_I2C: https://www.arduino.cc/reference/en/libraries/liquidcrystal-i2c/
- StackArray: https://github.com/elechouse/StackArray

Tutoriale și exemple:
https://circuitdigest.com/microcontroller-projects/arduino-calculator-using-4x4-keypad

Resurse Hardware

- Arduino: https://cleste.ro/kit-inva-are-arduino.h
- Keypad: https://cleste.ro/tasta-numerica-4x4.html
- LCD: https://www.robofun.ro/lcd/modul-afisaj-lcd-lumina-fundal-albastra-i2c.html

Export to PDF